**- ¿Cuáles son las bases de datos soportadas por Django y en qué se diferencian?**

Django admite varios motores de bases de datos, incluyendo PostgreSQL, MySQL, SQLite y Oracle. Cada uno tiene sus propias características y diferencias. Por ejemplo, PostgreSQL es conocido por ser altamente escalable y robusto, MySQL es popular por su rendimiento y facilidad de uso, SQLite es una base de datos liviana y portable, y Oracle es ampliamente utilizado en entornos empresariales.

**- ¿Qué es una migración en Django y para qué se utiliza?**

Una migración en Django es un mecanismo que permite realizar cambios en el esquema de la base de datos de forma controlada y mantenible. Se utiliza para crear, modificar o eliminar tablas, campos, índices, restricciones, entre otros elementos del esquema. Las migraciones garantizan que el esquema de la base de datos evolucione de manera coherente con los cambios realizados en el código.

**- ¿Cuál es la diferencia entre usar consultas SQL y consultas ORM en Django?**

La diferencia principal entre las consultas SQL y las consultas ORM en Django es el lenguaje utilizado para expresar las consultas. Las consultas SQL se escriben en lenguaje SQL, que es el estándar para interactuar con bases de datos relacionales. Las consultas ORM en Django se definen utilizando el lenguaje de consultas de objetos de Django, que permite interactuar con la base de datos utilizando el modelo de objetos de Django en lugar de escribir SQL directamente. Las consultas ORM proporcionan una capa de abstracción que facilita la escritura de consultas y mejora la portabilidad del código entre diferentes motores de bases de datos.

**- ¿Cómo se instalan los paquetes de base de datos en Django y cuál es su importancia?**

Los paquetes de base de datos en Django, como "psycopg2" para PostgreSQL o "mysqlclient" para MySQL, se instalan utilizando herramientas de gestión de paquetes de Python, como pip. Estos paquetes proporcionan la funcionalidad necesaria para que Django se comunique con la base de datos subyacente. Son importantes porque permiten la conexión y manipulación de datos en la base de datos específica elegida para el proyecto de Django.

**- ¿Qué ventajas ofrece Django como ORM para la integración con una base de datos?**

Django ofrece varias ventajas como ORM para la integración con una base de datos. Algunas de estas ventajas son:

* Abstracción del lenguaje SQL: Permite interactuar con la base de datos utilizando el lenguaje de consultas de objetos de Django en lugar de tener que escribir consultas SQL directamente.
* Portabilidad del código: El código escrito utilizando el ORM de Django es más portable entre diferentes motores de bases de datos, lo que facilita el cambio de base de datos sin tener que reescribir gran parte del código.
* Seguridad: El ORM de Django proporciona mecanismos para evitar ataques de inyección de SQL y otros problemas de seguridad comunes en consultas SQL directas.
* Facilidad de mantenimiento: Las migraciones en Django permiten gestionar de manera controlada y mantenible los cambios en el esquema de la base de datos a medida que evoluciona la aplicación.
* Funcionalidades adicionales: El ORM de Django ofrece funcionalidades adicionales, como la gestión de relaciones entre modelos, la validación de datos y la generación de consultas complejas de manera más sencilla.

[**https://github.com/MochitoSpooky/m7-individual-2.git**](https://github.com/MochitoSpooky/m7-individual-2.git)